PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-324361
(43)Date of publication of application : 07.12.1993

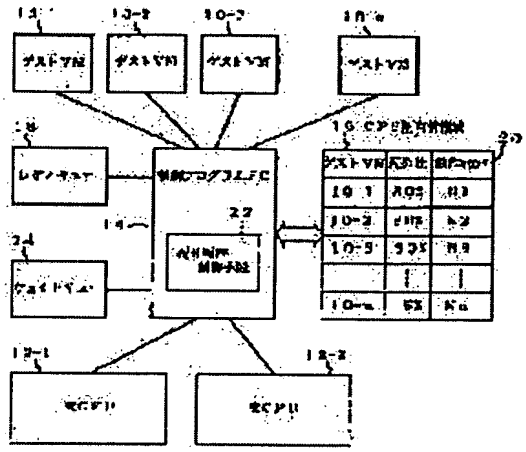
(51)Int.Cl. G06F 9/46

(21)Application number : 04-122985
(22)Date of filing : 15.05.1992

(71)Applicant : FUJITSU LTD
(72)Inventor : MATSUURA TAKEKAZU

(54) CPU CONTROL SYSTEM IN VIRTUAL COMPUTER SYSTEM

(57)Abstract:
PURPOSE: To maintain the distribution ratio of designated CPU resources and to efficiently utilize actual CPU even if the CPU distribution is unequal concerning the CPU control system of a virtual computer system where the CPU resources are distributed to plural virtual computers in the required ratio so as to permit the virtual computer to execute an operation.



CONSTITUTION: A system is provided in a CPU assignment information area 16, an operation counter 20 stores a time for digesting the distributed quantity of the designated CPU resources at every guest (VM10-1 to 10-n), the distribution order of the CPU resources in the guests VM10-1 to 10-n of a ready queue 18 is arranged in order of the operation times of the operation counter 20 from a small number by a distribution order control means 22, the CPU distribution of the guests VM10-1 to 10-n is executed in order of the distribution ratio from the large number when the operation times of the operation counter 20 are same and the CPU distribution of the guests VM10-1 to 10-n is executed in order of the operation times from the small number when the operation times are different.

LEGAL STATUS

[Date of request for examination]	27.12.1995
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	2682770
[Date of registration]	08.08.1997
[Number of appeal against examiner's decision of rejection]	

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平5－324361

(43)公開日　平成５年(1993)12月７日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　9/46	３５０	8120－5B		

審査請求　未請求　請求項の数６(全 17 頁)

(21)出願番号　特願平4－122985

(22)出願日　平成４年(1992)５月15日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中1015番地
(72)発明者　松浦　豪一
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内
(74)代理人　弁理士　竹内　進　(外１名)

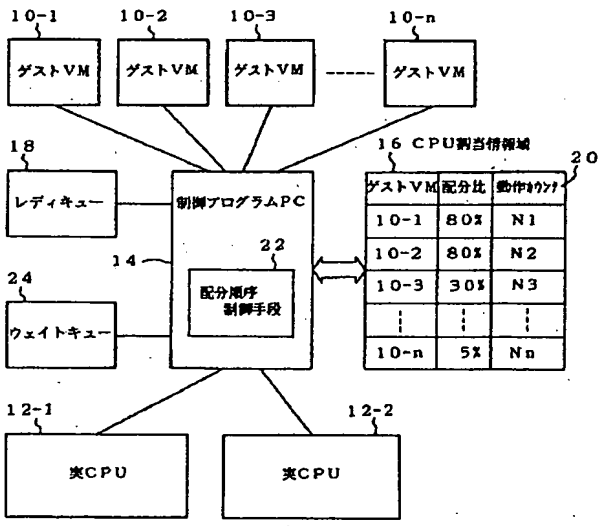
(54)【発明の名称】　仮想計算機システムのＣＰＵ制御方式

(57)【要約】

【目的】複数の仮想計算機に所望の割合でＣＰＵ資源を配分して仮想計算機を走行させる仮想計算機システムのＣＰＵ制御方式に関し、ＣＰＵ配分が不均等であっても、指定したＣＰＵ資源の配分比を維持して実ＣＰＵの有効利用を図る。

【構成】ＣＰＵ割当情報域１６に設けられ、指定されたＣＰＵ資源の配分量を消化した回数をゲストＶＭ１０－１～１０－ｎ毎に動作カウンタ２０で記憶し、レディキュー１８上のゲストＶＭ１０－１～１０－ｎのＣＰＵ資源の配分順序を配分順序制御手段２２により動作カウンタ２０の動作回数の小さい順番に並べ、動作カウンタ２０の動作回数が同一の場合は配分比の大きい順番にゲストＶＭ１０－１～１０－ｎのＣＰＵ配分を行い、動作回数が異なる場合には、動作回数の少ない順番にゲストＶＭ１０－１～１０－ｎのＣＰＵ配分を行う。

本発明の原理説明図



1

【 特許請求の範囲】

【 請求項1 】複数の実ＣＰＵ(１２－１，１２－２）を有する実計算機と、

該実計算機上で複数の仮想計算機(１０－１ ～１０－ｎ）を動作可能とする制御プログラム(１４）と、

前記仮想計算機(１０－１ ～１０－ｎ）毎に、割当てるべき前記実ＣＰＵ(１２－１，１２－２）の走行時間で決まるＣＰＵ資源の配分量を設定したＣＰＵ割当情報域(１６）と、

前記ＣＰＵ割当情報域(１６）との対応関係を示すポインタ情報を、前記仮想計算機(１０－１ ～１０－ｎ）のＣＰＵ資源の配分量が大きい程、高い優先順位をもつ配分順序となるように並べたレディキュー(１８）と、

を備え、前記制御プログラム(１４）によりレディキュー(１８）のポインタ情報で指定された順番に従って前記仮想計算機(１０－１ ～１０－ｎ）に実ＣＰＵ(１２－１，１２－２）を割当て、前記ＣＰＵ割当情報領域(１６）の配分量に従って繰り返し動作させる仮想計算機システムのＣＰＵ制御方式に於いて、

前記ＣＰＵ割当情報域(１６）に設けられ、指定されたＣＰＵ資源の配分量を消化した回数を仮想計算機(１０－１ ～１０－ｎ）毎に記憶する動作カウンタ(２０）と、

前記レディキュー(１８）上の仮想計算機(１０－１ ～１０－ｎ）のＣＰＵ資源の配分順序を前記動作カウンタ(２０）の動作回数の小さい順番に並べる配分順序制御手段(２２）と、

を備え、前記動作カウンタ(２０）の動作回数が同一の場合は配分比の大きい順番に仮想計算機(１０－１ ～１０－ｎ）のＣＰＵ配分を行い、動作回数が異なる場合には、動作回数の少ない順番に仮想計算機(１０－１ ～１０－ｎ）のＣＰＵ配分を行うことを特徴とする仮想計算機システムのＣＰＵ制御方式。

【 請求項2 】請求項1 記載の仮想計算機システムのＣＰＵ制御方式に於いて、ＣＰＵ資源の配分時に待ち状態にある仮想計算機(１０－ｉ ）のＣＰＵ割当情報域(１６）を示すポインタ情報を取付けるウェイトキュー(２４）を設け、前記制御プログラム(１４）は前記仮想計算機(１０－ｉ ）が動作可能状態となった時に前記ウェイトキュー(２４）のポインタ情報を外して前記ウェイトキュー(１８）に取付けることを特徴とする仮想計算機システムのＣＰＵ制御方式。

【 請求項3 】請求項2 記載の仮想計算機システムのＣＰＵ制御方式に於いて、前記制御プログラム(１４）は、仮想計算機(１０－ｉ ）がＣＰＵ資源配分中に待ち状態になったことを検出した場合、該仮想計算機(１０－ｉ ）が使い切っていない残り資源量をＣＰＵ割当用情報域(１６）の対応位置に格納すると共に前記ウェイトキュー(２４）に対応するＣＰＵ割当情報域(１６）を示すポインタ情報を取付け、待ち状態の仮想計算機(１０－ｉ ）が動作可能になったことを検出した場合には、ポインタ情報を前記ウェイトキュー(２４）から外して前記レディキュー(１８）の前記動作カウンタ(２０）の動作回数と配分比の大きさで決まる優先順位に従った位置に移してＣＰＵ資源の配分を再開させることを特徴とする仮想計算機システムのＣＰＵ制御方式。

2

【 請求項4 】請求項1 記載の仮想計算機システムのＣＰＵ制御方式に於いて、前記配分順序制御手段(２２）は、特定の仮想計算機(１０－ｉ ）に対するＣＰＵ資源の配分が終了する毎に、前記動作カウンタ(２０）の動作回数と各仮想計算機(１０－１ ～１０－ｎ）の配分比を調べ、動作回数が同一の場合は配分比の大きい順番に並べ、動作回数が異なっている場合には、動作回数の少ないものを先頭側に移して配分比の大きい順番に並べ、次に動作回数の多いものを配分比の大きい順番に並べることを特徴とする仮想計算機システムのＣＰＵ制御方式。

【 請求項5 】請求項1 記載の仮想計算機システムのＣＰＵ制御方式に於いて、前記動作カウンタ(２０）は、ＣＰＵ資源の配分量で規定された時間だけ仮想計算機(１０－１ ～１０－ｎ）が走行する毎に1 つカウントアップされ、該走行が完了した仮想計算機(１０－ｉ ）を示すポインタ情報を前記配分順序制御手段(２２）に従ったレディキュー(１６）の位置に移すことを特徴とする仮想計算機システムのＣＰＵ制御方式。

【 請求項6 】請求項1 記載の仮想計算機システムのＣＰＵ制御方式に於いて、前記レディキュー(１８）には、現時点で優先順位が最も高い仮想計算機のＣＰＵ割当情報域(１６）を示す先頭ポインタ情報と、現時点で優先順位が最も低い仮想計算機のＣＰＵ割当情報域(１６）を示す末尾ポインタ情報とを格納し、先頭ポインタ情報で指定された仮想計算機のＣＰＵ資源の配分が終了する毎に、先頭ポインタ情報を末尾ポンインタ情報の位置に移すことを特徴とする仮想計算機システムのＣＰＵ制御方式。

【 発明の詳細な説明】

【 ０００１ 】

【 産業上の利用分野】本発明は、複数のＣＰＵを有する実計算機上で複数の仮想計算機を動作可能とする制御プログラムによる仮想計算機システムのＣＰＵ制御方式に関し、特に、複数の仮想計算機に所望の割合でＣＰＵ資源を配分して仮想計算機を走行させる仮想計算機システムのＣＰＵ制御方式に関する。

【 ０００２ 】仮想計算機モニタプログラム(ＶＭモニタ）として知られた制御プログラムＣＰにより実計算機上で複数の仮想計算機(以下「 ゲストＶＭ」という）を動作可能とし、ゲストＶＭの各々がオペレーティングシステムＯＳ で動作できる仮想計算機システムが大型コンピュータの利用技術の一つとして知られている。このような仮想計算機システムで、複数のゲストＶＭを複数の

3

ＣＰＵ上で動作させる場合には、ゲストＶＭがＣＰＵ能力を有効に利用できるようにするため、各ゲストＶＭにＣＰＵ資源の配分量（配分比）を決め、この配分量に従って順番にゲストＶＭを走行させている。ゲストＶＭの走行順は、ＣＰＵ資源の割当量の多い順番となる。

【０００３】しかし、配分量が均等でなかった場合には、ゲストＶＭの走行を繰り返すうちに、割当量の少ないゲストＶＭの走行回数が増加するようになり、意図したＣＰＵ資源の配分量が達成できなくなり、この点の改善が望まれる。

【０００４】

【従来の技術】図9は従来の仮想計算機システムを示したもので、制御プログラム（ＣＰ）１４によって実ＣＰＵ１２を備えた実計算機上に複数のゲストＶＭ１０－１～１０－ｎを動作可能としており、ゲストＶＭ１０－１～１０－ｎはオペレーティングシステムＯＳを各々動作させることができる。

【０００５】この場合、実計算機システムは単一のＣＰＵ１２のみを備えており、予め定めた配分比に従ってゲストＶＭ１０－１～１０－ｎに実ＣＰＵ１２を割当て順番に走行させている。制御プログラム１４に対しては、ＣＰＵ割当情報領域として処理制御ブロック（ＰＣＢ；Process Control Block）１６が設けられ、ゲストＶＭ１０－１～１０－ｎ毎に予め定めたＣＰＵ資源の配分比が格納されいてる。

【０００６】ゲストＶＭ１０－１～１０－ｎの走行順位は、レディキュー１８により決められる。レディキュー１８にはゲストＶＭ１０－１～１０－ｎに対するＣＰＵ資源の配分順序が、配分比の大きい順番に従って設定されている。レディキュー１８には、制御処理ブロック１６のゲストＶＭ１０－１～１０－ｎを指定するポインタ情報がＣＰＵ配分比で決まる優先順位に従って格納される。

【０００７】実際には、現在時点で最も優先順位の高い先頭ポインタ情報と最も優先順位の低い末尾ポインタ情報が格納されている。このため制御プログラム１４は実ＣＰＵ１２が動作可能となった時に、レディキュー１８の先頭ポインタで指定される処理制御ブロック１６を参照して、ＣＰＵを割当てるゲストＶＭと配分比を知り、配分比で規定される時間に亘り実ＣＰＵ１２により指定されたゲストＶＭでＯＳを走行させる。

【０００８】更にウェイトキュー24が設けられ、レディキュー１８の優先順位に従ってＣＰＵ配分を受けたゲストＶＭが待ち状態にあるとき、このゲストＶＭをウェイトキュー２４に取付け、次のゲストＶＭにＣＰＵ配分を行う。待ち状態にあったゲストＶＭが動作可能になると、ウェイトキュー２４から外してレディキュー１８の末尾ポインタ位置に取り付けてＣＰＵ配分を行う。

【０００９】図１０は図9の仮想計算機システムの配分処理を示したフローチャートであり、まずステップＳ１

4

でゲストＶＭ１０－１～１０－ｎのレディキュー１８へのポインタ情報の取り付けをＣＰＵ配分比で決まる順番に従って行ない、先頭ポインタ情報による処理制御ブロック１６の内容から指定されるゲストＶＭを選択する。

【００１０】続いてステップＳ２に進んで、レディキュー１８に基づく優先順位に従って選択したゲストＶＭに実ＣＰＵ１２を配分し、配分比で規定される時間に亘り実ＣＰＵ１２に選択したゲストＶＭの動作を行わせる。ステップＳ２の配分済むと、ステップＳ３で配分が済んだゲストＶＭのポインタ情報を末尾ポインタ情報の位置に移し、次に優先順位の高かったポインタ情報が先頭ポインタ情報の位置に移し、再びステップＳ１に戻って次のゲストＶＭの動作を行う。

【００１１】図１１は4つのゲストＶＭを対象に単一の実ＣＰＵでＣＰＵ配分を行った場合の処理順を示す。尚、４つのゲストＶＭの名称をＶＭ１，ＶＭ２，ＶＭ３，ＶＭ4として示している。ここでゲストＶＭ１～ＶＭ4のＣＰＵ資源の配分比は、処理制御ブロック１６により

ＶＭ１	８０％
ＶＭ２	１０％
ＶＭ３	５％
ＶＭ４	５％

に予め設定されているものとする。

【００１２】単一の実ＣＰＵで、複数のゲストＶＭ１～４にＣＰＵ資源を配分する場合は、ＣＰＵ資源の配分の順番をＣＰＵ配分比の大きい順番に図１１のように設定し、指定された順序に従い指定された通りの配分比を得ることができる。

【００１３】

【発明が解決しようとする課題】このような従来の仮想計算機システムにあっては、図9に示したように基本的には単一のＣＰＵ１２の場合にのみ指定された配分比に従って厳密にＣＰＵ資源を分配することが可能である。しかしながら、図１２のような複数の実ＣＰＵ１２－１，１２－2の場合には、指定された配分比に従って厳密にＣＰＵ資源を分配することができないという問題があった。

【００１４】これは処理制御ブロック１６を繋げるレディキュー１８における優先順位が配分比の大きさによってのみ並べられていたためである。単一のＣＰＵの場合は、配分比に従って並べることで、指定した配分比でＣＰＵ資源が分配できるが、複数のＣＰＵの場合は、より小さい配分比のゲストＶＭがより大きい配分比のゲストＶＭより多くの回数ＣＰＵ配分を受けることがあり、指定した配分比が得られなくなる。これを詳細に説明すると次のようになる。

【００１５】まず図１２の複数の実ＣＰＵ１２－１，１２－2の場合に、図9の単一の実ＣＰＵ１２の場合と同様、配分比の大きさで決まる優先順位に従って複数のゲ

5

ストＶＭ１０－１～１０－ｎにＣＰＵ資源の配分を行うと、ＣＰＵ資源の配分比が全て同じで且つ実ＣＰＵに対する割当てが均等になっている場合はＣＰＵ配分どおりに配分できる。

【００１６】図１３は実ＣＰＵ１２－１にゲストＶＭ１，ＶＭ３，ＶＭ５，ＶＭ７の４つを割当て、実ＣＰＵ１２－２にゲストＶＭ２，ＶＭ４，ＶＭ６，ＶＭ８の４つを割当てる均等割当てを行い、且つ番号の若い順番に優先順位を設定した場合であり、ゲストＶＭ１～８に対するＣＰＵ配分の順序性が保たれたまま処理を繰り返すことができる。

【００１７】しかし、ＣＰＵ配分比が不均等な場合には、処理を繰り返すにつれて最初に指定したＣＰＵ配分の順序性が崩れ、指定したＣＰＵ資源の配分比が実現できなくなる。図１４は５つのゲストＶＭ１～５を対象に２台の実ＣＰＵ１２－１，１２－２のＣＰＵ資源の配分処理を示したもので、ゲストＶＭ１～５の配分比は不均等であり、

ＶＭ１　８０％
ＶＭ２　８０％
ＶＭ３　３０％
ＶＭ４　５％
ＶＭ５　５％

に定められているものとする。

【００１８】最初のＣＰＵ配分の順序は配分比の大きい順に、

ＶＭ１→ＶＭ２→ＶＭ３→ＶＭ４→ＶＭ５

となっており、１回目は、この順番に従って実ＣＰＵ１２－１，１２－２のＣＰＵ資源の配分が行われる。即ち、実ＣＰＵ１２－１をゲストＶＭ１に割当てると同時に、実ＣＰＵ１２－２をゲストＶＭ２に割当てて処理を開始する。ゲストＶＭ１とゲストＶＭ２の配分比は共に８０％と同じなので、この配分比８０％で決まる割当時間終了の同時刻でゲストＶＭ１とゲストＶＭ２の配分を終る。

【００１９】次に実ＣＰＵ１２－１をゲストＶＭ３に割当て、同時に実ＣＰＵ１２－２をゲストＶＭ４に割当てる。ゲストＶＭ３の配分比は３０％であり、ゲストＶＭ４の配分比は５％であることから、先にゲストＶＭ４の配分が終了し、実ＣＰＵ１２－２を最後のゲストＶＭ５に割当てる。これが１回目のＣＰＵ配分である。ここでゲストＶＭ４及びゲストＶＭ５が配分を終了してもゲストＶＭ３は以前として配分中にあり、このため先に配分が終ったゲストＶＭ４とゲストＶＭ５がレディキュー１８に順序付けられてしまう。

【００２０】２回数目のＣＰＵ配分は、１回目のゲストＶＭ５の配分が終了した時点で実ＣＰＵ１２－２をゲストＶＭ１に割当てることで開始される。このとき実ＣＰＵ１２－１による１回目のゲストＶＭ３の配分はまだ終了していない。１回目のゲストＶＭ３の配分が終了すると、実ＣＰＵ１２－１をゲストＶＭ２に割当てて２回目のＣＰＵ配分を行う。

6

【００２１】このゲストＶＭ３の１回目の配分が終了した時点で確定した２回目の配分順序は、

ＶＭ１→ＶＭ２→ＶＭ４→ＶＭ５→ＶＭ３

となっている。

【００２２】同様にして２回目の配分を終了した時点で確定した３回目の配分順序は、

ＶＭ１→ＶＭ４→ＶＭ５→ＶＭ３→ＶＭ２

となる。

【００２３】図１５は、図１４のＣＰＵ配分の様子を時間割合で示したもので、実ＣＰＵ１２－１，１２－２に対するゲストＶＭ１～５の時間的な割当てが判る。このように実ＣＰＵ１２－１，１２－２に対して不均等な配分比をもつゲストＶＭ１～５の配分処理を行うと、配分比の小さいゲストＶＭ４やゲストＶＭ５は配分比の大きいゲストＶＭ３へのＣＰＵ配分が完了する前にＣＰＵ配分が完了してしまうため、次回の配分処理では、ゲストＶＭ３よりも早くＣＰＵが配分され、このような処理を繰り返すと指定された配分どおりにＣＰＵを配分できなくなる。

【００２４】図１６は図１４のＣＰＵ配分を繰り返して飽和状態に至った時のＣＰＵ配分の指定量に対する実際の割当量を示したもので、指定量の少ないゲストＶＭ４，ＶＭ５の割当量が増加し、指定量の大きいゲストＶＭ１～ＶＭ３は割当量が少なめになる。一方、ＣＰＵ配分が図１３に示したように均等だったとしても、ゲストＶＭが走行中に待ち状態となった場合には、制御プログラム１４が待ち状態にあるゲストＶＭの制御を取り上げてウェイトキュー２４に取付け、次のゲストＶＭにＣＰＵ配分を行う。

【００２５】ウェイトキュー２４に取付けゲストＶＭが動作可能になると、制御プログラム１４は、レディキュー１８の末尾ポインタン位置に動作可能となったゲストＶＭを取付ける。このためゲストＶＭの待ち状態が生ずると、実ＣＰＵに対して不均等なＣＰＵ配分が一時的に発生し、指定された配分どおりにＣＰＵを配分できなくなる。

【００２６】本発明は、このような従来の問題点に鑑みてなされたもので、ＣＰＵ配分が不均等であっても、指定したＣＰＵ資源の配分比を維持して実ＣＰＵの有効利用を図るようにした仮想計算機システムのＣＰＵ制御方式を提供することを目的とする。また本発明は、均等なＣＰＵ配分に従った処理途中でゲストＶＭに処理待ちが発生しても、指定したＣＰＵ資源の配分比を維持して実ＣＰＵの有効利用を図るようにした仮想計算機システムのＣＰＵ制御方式を提供することを目的とする。

【００２７】

【課題を解決するための手段】図１は本発明の原理説明図である。まず本発明は、複数の実ＣＰＵ１２－１，１

２－２を有する実計算機と、実計算機上で複数のゲストＶＭ(仮想計算機) １０－１～１０－ｎを動作可能とする制御プログラム(ＣＰ) １４と、ゲストＶＭ１０－１～１０－ｎ毎に、割当てるべき実ＣＰＵ１２－１，１２－２の走行時間で決まるＣＰＵ資源の配分量を設定したＣＰＵ割当情報域(ＰＣＢ) １６と、ＣＰＵ割当情報域１６との対応関係を示すポインタ情報を、ゲストＶＭ１０－１～１０－ｎのＣＰＵ資源の配分量が大きい程、高い優先順位をもつ配分順序となるように並べたレディキュー１８とを備え、制御プログラム１４によりレディキュー１８のポインタ情報で指定された順番に従ってゲストＶＭ１０－１～１０－ｎに実ＣＰＵ１２－１，１２－２を割当て、ＣＰＵ割当情報領域１６の配分量に従って繰り返し動作させる仮想計算機システムのＣＰＵ制御方式を対象とする。

【００２８】このような仮想計算機システムのＣＰＵ制御方式につき本発明にあっては、ＣＰＵ割当情報域１６に設けられ、指定されたＣＰＵ資源の配分量を消化した回数をゲストＶＭ１０－１～１０－ｎ毎に記憶する動作カウンタ２０と、レディキュー１８上のゲストＶＭ１０－１～１０－ｎのＣＰＵ資源の配分順序を動作カウンタ２０の動作回数の小さい順番に並べる配分順序制御手段２２とを備え、動作カウンタ２０の動作回数が同一の場合は配分比の大きい順番にゲストＶＭ１０－１～１０－ｎのＣＰＵ配分を行い、動作回数が異なる場合には、動作回数の少ない順番にゲストＶＭ１０－１～１０－ｎのＣＰＵ配分を行うことを特徴とする。

【００２９】更に、ＣＰＵ資源の配分時に待ち状態にある仮想計算機１０－ｉのＣＰＵ割当情報領域１６を示すポインタ情報を取付けるウェイトキュー２４を設け、制御プログラム１４はゲストＶＭ１０－ｉが動作可能状態となった時にウェイトキュー２４のポインタ情報を外してウェイトキュー(１８) に取付ける。また仮想計算機１０－ｉがＣＰＵ資源配分中に待ち状態になったことを制御プログラム１４が検出した場合には、待ち状態となったゲストＶＭ１０－ｉが使い切っていない残り資源量(残り時間) をＣＰＵ割当用情報域１６の対応位置に格納すると共にウェイトキュー２４に対応するＣＰＵ割当情報域１６を示すポインタ情報を取付け、待ち状態の仮想計算機１０－ｉが動作可能になったことを検出した場合には、ポインタ情報をウェイトキュー２４から外してレディキュー１８の動作カウンタ２０の動作回数と配分比の大きさで決まる優先順位に従った位置に移してＣＰＵ資源の配分を再開させる。

【００３０】一方、制御プログラム１４の配分順序制御手段２２は、特定の仮想計算機１０－ｉに対するＣＰＵ資源の配分が終了する毎に、動作カウンタ２０の動作回数と各ゲストＶＭ１０－１～１０－ｎの配分比を調べ、動作回数が同一の場合は配分比の大きい順番に並べ、動作回数が異なっている場合には、動作回数の少ないものを先頭側に移して配分比の大きい順番に並べ、次に動作回数の多いものを配分比の大きい順番に並べる。

【００３１】また動作カウンタ２０は、ＣＰＵ資源の配分量で規定された時間だけゲストＶＭ１０－１～１０－ｎが走行する毎に1つカウントアップされ、走行が終了したゲストＶＭを示すポインタ情報を配分順序制御手段２２に従ったレディキュー１６の位置に移す。さらにレディキュー１８には、現時点で優先順位が最も高いゲストＶＭのＣＰＵ割当情報領域１６を示す先頭ポインタ情報と、現時点で優先順位が最も低いゲストＶＭのＣＰＵ割当情報領域１６を示す末尾ポインタ情報とを格納し、先頭ポインタ情報で指定されたゲストＶＭのＣＰＵ資源の割当量の配分が終了する毎に、先頭ポインタ情報を末尾ポンインタ情報の位置に移す。

【００３２】

【作用】このような構成を備えた本発明の仮想計算機システムのＣＰＵ制御方式によれば、複数のゲストＶＭついて予め定めたＣＰＵ資源の配分量(配分比) が不均等であっても、ＣＰＵ配分の順序をゲストＶＭの動作回数と配分比の両方で決めているため、配分処理を繰り返しても最初に設定した順序性を維持することができ、複数の実ＣＰＵの割当てにより指定されたＣＰＵ配分通りに実際の配分ができ、実ＣＰＵ資源を有効に利用できる。

【００３３】例えば配分比の小さいゲストＶＭが配分を繰り返す内に優先順位が高くなり、配分比の大きなゲストＶＭの優先順位が低くなってきても、動作回数を求めて動作回数の少ないゲストＶＭを優先させるため、同じ動作回数となる全てのゲストＶＭの配分が終了しない限り、次の動作回数でのＣＰＵ配分には移行せず、配分比の少ないゲストＶＭの動作回数が配分比の大きなゲストＶＭの動作回数より多くなってしまうことを防止できる。

【００３４】またＣＰＵ配分が均等な場合に、ＣＰＵ配分中にゲストＶＭに待ち状態が起きて一時的にＣＰＵ配分が不均等になっても、動作可能状態となった場合には、動作回数と配分比に従った順番の配分位置に取付けられるため、次の動作回数に段階では不均等状態が解消され、指定した配分比を維持できる。

【００３５】

【実施例】図2は本発明の仮想計算機のＣＰＵ制御方式が適用されるハードウェア構成を示した説明図である。図2において、１２－１，１２－２は実計算機を構成するＣＰＵ(「実ＣＰＵ」ともいう) であり、主記憶制御装置(ＭＣＵ) ２６を介して主記憶装置(ＭＳＵ) ２８に接続している。主記憶装置２８にはオペレーティングシステム(ＯＳ) ３０が設けられ、実計算機システムとしてはオペレーティングシステム３０を実計算機１２－１，１２－２で実行する。

【００３６】また、主記憶装置２８には仮想計算機システムを実現するためＶＭモニタとして知られた制御プロ

9

グラム（ＣＰ）１４が設けられている。制御プログラム１４は実計算機システム上で仮想計算機としてのゲストＶＭ１０－１，１０－２，・・・１０－ｎを動作可能とし、制御プログラム１４で動作可能となったゲストＶＭ１０－１～１０－ｎのそれぞれはオペレーティングシステム３０を走行させることができる。

【００３７】また、主記憶制御装置２６にはチャネルプロセッサ（ＣＨＰ）３２を介してデバイスバス３４が接続され、デバイスバス３４には複数のデバイスインタフェース３６－１，３６－２を設けている。この実施例では、デバイスインタフェース３６－１に磁気ディスク装置３８を接続している。図3は図2のハードウェア構成で実現される本発明の仮想計算機システムの一実施例を示した説明図である。

【００３８】図3にあっては、制御プログラム１４により５つのゲストＶＭ１０－１～１０－５を実現して動作可能とする仮想計算機システムを例にとっている。制御プログラム１４に対してはＣＰＵ割当て情報域として処理制御ブロック（ＰＣＢ）１６が設けられる。処理制御ブロック１６はゲストＶＭの名称に対応して予め定めたＣＰＵ配分の配分比を設定しており、更に本発明にあっては、新たに動作カウタン２０を設けている。

【００３９】ここで、ゲストＶＭ１０－１～１０－５の名称は、説明を簡単にするためＶＭ1～ＶＭ5として示す。また、ＣＰＵ資源の配分比はパーセント表示としている。ＣＰＵ配分比に従ったＣＰＵ資源の配分は実際には実ＣＰＵ１２－１または１２－２の割当て時間となる。動作カウンタ２０はゲストＶＭ１０－１～１０－５に設定されたＣＰＵ資源の配分比に従った実ＣＰＵ１２－１または１２－２の割り当てを受けて配分処理を１回消費するごとに1つカウントアップした値を動作回数として記憶する。この実施例にあっては動作カウンタ２０の動作回数をゲストＶＭ１０－１～１０－５に対応して動作回数Ｎ１～Ｎ５で示している。

【００４０】更に、制御プログラム１４に対してはレディキュー１８とウェイトキュー２４が設けられる。レディキュー１８には制御プログラム１４に設けた本発明による配分順序制御手段２２で定めたゲストＶＭ１０－１～5に対するＣＰＵ資源の配分順序に従った情報が取り付けられる。具体的にレディキュー１８には処理制御ブロック１６のゲストＶＭ１０－１～5を示すポインタ情報が取り付けられる。

【００４１】配分順序制御手段２２によるゲストＶＭ１０－１～5の決定ルールは次のルール1及びルール2に従う。

［ルール1］；ゲストＶＭ１０－１～5の動作カウンタの動作回数Ｎ１～Ｎ５の小さいものを優先する。

【００４２】［ルール2］；ゲストＶＭ１０－１～5の中で配分比の大きい順番に並べる。

ここで、配分順序を決めるルール1とルール2の関係は

10

ルール1の方が優先度が高く、ルール1を満足した状態でルール2が適用される。具体的には、動作カウンタの動作回数Ｎ１～Ｎ５が同じであれば配分比の大きい順にレディキュー１８にポインタ情報を取り付ける。

【００４３】また、ゲストＶＭ１０－１～１０－５の動作カウンタ２０の動作回数Ｎ１～Ｎ５が異なっている、例えばいずれか1つの動作カウンタ２０の動作回数が他の動作回数より少なかった場合には動作回数の少ないゲストＶＭをレディキュー１８の先頭に取り付け、動作回数の少ないゲストＶＭの割当てを優先する。レディキュー１８の具体的な構成としては、配分順序制御手段２２により前記ルール1，２により決定されたゲストＶＭ1～5のＣＰＵ配分の順番について、現時点で先頭に位置するゲストＶＭの先頭ポインタ情報と現時点で最後に位置するゲストＶＭの末尾ポインタ情報の2つを格納する。

【００４４】そして、先頭ポインタで指定されるゲストＶＭに対するＣＰＵ配分が終了すると、先頭ポインタを外して末尾ポインタに移し、次に高い順位にあるゲストＶＭのポインタ情報を先頭ポインタに移し、先頭ポインタで指定されたゲストＶＭに対するＣＰＵ配分を行い、これをシーケンシャルに繰り返す。一方、ウェイトキュー２４は制御プログラム１４により新たなゲストＶＭにＣＰＵ配分を行う際に、このゲストＶＭが待ち状態にあるとき、この待ち状態にあるゲストＶＭのポインタ情報をレディキュー１８の先頭ポインタ位置から外してウェイトキュー２４に取り付ける。

【００４５】このウェイトキュー２４へのポインタ情報の取り付けによりレディキュー１８の先頭ポインタ位置には次に優先順位の高いゲストＶＭのポインタ情報がくることから、次のゲストＶＭに対しＣＰＵ配分を行う。勿論、ウェイトキュー２４には複数のゲストＶＭのポインタを取り付けることができ、最大5つのゲストＶＭ１０－１～5分の取付領域があればよい。

【００４６】ウェイトキュー２４に対し待ち状態にあるゲストＶＭのポインタ情報を取り付けた後、制御プログラム１４は待ち状態にあるゲストＶＭが動作可能になるか否か常時監視しており、動作可能になるとウェイトキュー２４からポインタ情報を取り外し、配分順序制御手段２２のルール1及びルール2で決まる優先順位の位置となるようにレディキュー１８にウェイトキュー２４から取り外したポインタ情報を取り付ける。

【００４７】一方、制御プログラム１４によりＣＰＵ割当てを受け、配分中にゲストＶＭが待ち状態となった場合には、制御プログラム１４はウェイトキュー２４にレディキューの先頭ポインタからポインタ情報を外して取り付けると同時に処理制御ブロック１６の対応するゲストＶＭの配分比を残りの配分比となるように処理する。

【００４８】このため、動作可能状態となって再度レディキュー１８に取り付けられてＣＰＵ配分を受けた場合

11

には、待ち状態となったときに残っている配分比分だけの再配分を行うことになる。図4は図3の仮想計算機システムにおけるＣＰＵ配分処理を示したフローチャートである。

【００４９】図4において、まずステップＳ１で配分順序制御手段２２がその時の処理制御ブロック１６における動作カウンタ２０の動作回数Ｎ１～Ｎ５と配分比を調べ、ゲストＶＭ１０－１～5の配分順序を決定し、レディキュー１８に最初と最後のゲストＶＭのポインタ情報を取り付けており、このレディキュー１８の先頭ポインタ情報を読み出してＣＰＵ配分を行うゲストＶＭを選択する。

【００５０】続いてステップＳ２で選択したゲストＶＭにこのとき空き状態にある実ＣＰＵ１２－１または１２－２を割当て、ＣＰＵ配分を開始する。このＣＰＵ配分の実行中にあっては、ステップＳ３で現在ＣＰＵ配分を受けているゲストＶＭの状態を監視しており、割当てられた実ＣＰＵ１２－１または１２－２による処理時間が指定された配分比で規定される時間だけ使用されると、配分完了としてステップＳ４に進む。

【００５１】ステップＳ４にあっては、動作カウンタ２０の対応する動作回数に1を足す。続いてステップＳ５に進み、配分順序制御手段２２がルール1及び2に従った優先度に従ってレディキュー１８に対する割当て順番の再度取付けを行う。続いてステップＳ６に進み、後の説明で明らかにする待ち状態にあるゲストＶＭの中に動作可能なものがあるか否かチェックし、このとき待ち状態にあるゲストＶＭはないことから再びステップＳ１に戻り、レディキューから次のゲストＶＭを取り出して同様にＣＰＵ配分を行う。

【００５２】一方、ステップＳ３でＣＰＵ配分を受けたゲストＶＭが配分比または配分処理の途中で待ち状態となった場合にはステップＳ８に進み、待ち状態における残り配分比、即ち残りの割当て時間を対応する処理制御ブロック（ＰＣＢ）１６の配分比の部分に格納する。続いてステップＳ９でウェイトキュー２４に待ち状態となったゲストＶＭのポインタ情報を取り付ける。

【００５３】ウェイトキュー２４に対するポインタ情報の取り付けが済むとステップＳ５で配分順序制御手段２２がルール1及びルール2に従って配分順序の再取付けを行い、ステップＳ６でウェイトキュー２４に取付けたゲストＶＭの中で動作可能となったゲストＶＭがあるか否かチェックする。このとき、動作可能となったゲストＶＭがあった場合にはステップＳ７に進み、ウェイトキュー２４のポインタ情報を取り外し、配分順序制御手段２２のルール1及びルール2に従った優先度に従ってレディキュー１８にウェイトキュー２４から取り外したゲストＶＭのポインタ情報を取り付け、待ち状態から動作可能状態となったゲストＶＭに対するＣＰＵ配分の再開を可能とする。

12

【００５４】図5は図3の実施例に示した本発明によるＣＰＵ配分の順番を示した説明図であり、図6に図5のＣＰＵ配分をＣＰＵ割当て時間の説明図として示す。図5のＣＰＵ配分処理はゲストＶＭ１０－１～5の名称をＶＭ1～ＶＭ5で表し、配分比を

ＶＭ1　８０％
ＶＭ2　８０％
ＶＭ3　３０％
ＶＭ4　５％
ＶＭ5　５％

に設定した場合である。初期状態において、動作カウンタ２０のゲストＶＭ1～ＶＭ5の動作回数Ｎ１～Ｎ５はそれぞれＮ１～Ｎ５＝1となっている。従って、配分順序制御手段２２はルール2に従った配分順序を設定する。即ち、ゲストＶＭ1～ＶＭ5の配分比に従い、図7に示すように、配分比の大きい順番に並べる。

【００５５】１回目のＣＰＵ配分の動作は実ＣＰＵ１２－２を優先順位の最も高いゲストＶＭ1に割り付け、同時に実ＣＰＵ１２－２を次に優先順位の高いゲストＶＭ２に割り付け、ゲストＶＭ1とＶＭ2の配分処理を同時に開始する。ゲストＶＭ1とゲストＶＭ2の配分比は同じ８０％であることから、図6から明らかなように実ＣＰＵ１２－１，１２－２によるゲストＶＭ1，ＶＭ2の配分処理は同時刻に終了する。

【００５６】続いて実ＣＰＵ１２－１を3番目に優先度の高いゲストＶＭ3に割り付け、同時に実ＣＰＵ１２－２を4番目に優先度の高いゲストＶＭ4に割り付け、同時に配分処理を開始する。実ＣＰＵ１２－２を割り付けたゲストＶＭ4の配分比は5％と小さいため、実ＣＰＵ１２－１を割り付けた配分比３０％のゲストＶＭ3の配分処理が終了する前にゲストＶＭ4の配分処理が終了する。ゲストＶＭ4の配分処理が終了すると実ＣＰＵ１２－２をゲストＶＭ5に割り付け、ゲストＶＭ5の配分処理を開始する。ゲストＶＭ5の配分処理が終了しても、このときゲストＶＭ3はまだ配分処理中にある。

【００５７】このため、ゲストＶＭ5の配分処理が終了すると実ＣＰＵ１２－２をゲストＶＭ1に割り付け、２回目の配分処理を開始し、ゲストＶＭ1の配分処理中にゲストＶＭ3の1回目の配分処理が終了する。１回目のゲストＶＭ1～ＶＭ5の配分処理が終了するごとに動作カウンタ２０の動作回数Ｎ１～Ｎ５は1つずつカウントアップされ、次のＣＰＵ配分を開始する際に、配分順序制御手段２２がルール1及び2に従った優先順位に基づく配分順の設定を行う。

【００５８】このため、第1回目の配分処理において、ゲストＶＭ4とＶＭ5の配分処理がゲストＶＭ3の配分処理より前に終了していても、最後にゲストＶＭ3の配分が終了してゲストＶＭ1～3の1回目の配分処理が終了した段階で動作カウンタ２０の動作回数Ｎ１～Ｎ５は全て同じ値となっており、動作回数Ｎ１～Ｎ５が同じ場

13

合にはルール2 による配分比の大きい順に並べることから、２回目の配分処理を開始した段階でも１回目と同様、ゲストＶM1 ～ＶM5 の配分順位は図7 に示した最初と同じ状態を維持する。

【００５９】従って、配分処理を繰り返しても処理制御ブロック１６で指定した配分比を維持することができ、実ＣＰＵ１２－１，１２－2 を効率よく利用することができる。図8 はゲストＶM1 ～5 の配分処理の途中でゲストＶM2 に待ち状態が生じたときの説明図である。

【００６０】図8 において、１回目の配分処理でゲストＶM1 ～ＶM5 の順番で配分処理を行っているとき、ゲストＶM2 が配分比１０％に相当する時間、配分処理を行ったときに待ち状態を生じたとする。このようにゲストＶM2 が待ち状態になるとそのポインタ情報がウェイトキュー２４に取り付けられ、同時に制御処理ブロック１６のゲストＶM2 の配分比を残り７０％に設定する。

【００６１】ゲストＶM2 の待ち状態への以降処理が済むと次のゲストＶM3 の配分処理に移行する。１回目の配分処理がゲストＶM5 間で済んだ状態で待ち状態にあったゲストＶM2 が動作可能状態になったとすると、そのポインタ情報をウェイトキュー２４から取り外してレディキュー１８に取り付ける。このとき、レディキューにはすでに1 回目の割り付け処理が済んで2 回目の割り付け処理待ち状態にあるゲストＶM1 及びＶM3 ～ＶM5 が並んでいるが、待ち状態におかれたゲストＶMの動作回数Ｎ２はＮ２＝1 と２回目の配分処理を待っているゲストＶM1 およびＶM3 の動作回数Ｎ１，Ｎ３～Ｎ５＝2 より小さいいことから、動作可能状態となったゲストＶM2 の残りの配分比７０％の配分処理が優先して行われる。

【００６２】このため、待ち状態にあったゲストＶM2 の1 回目の配分処理が完了すると２回目の割り付け処理の順番は全ての動作回数Ｎ１～Ｎ２＝2 となっていることから、配分比の大きい順番に並べられ、図7 に示した順番を維持した配分処理が繰り返される。このため、特定のゲストＶMが待ち状態となって一時的に配分順序が乱れても、次の配分処理の際には正しい順番に戻った配分が行われることになる。

【００６３】尚、上記の実施例は2 つの実ＣＰＵを5 つのゲストＶMに割り付けてＣＰＵ資源の配分処理を行う場合を例にとるものであったが、本発明はこれに限定されず、任意の数の実ＣＰＵを同じく任意の数のゲストＶMに割り付ける配分処理にそのまま適用することができる。

【００６４】

【発明の効果】以上説明してきたように本発明によれば、指定された配分量を消費するごとに1 つカウントアップされる動作カウンタの動作回数を用いた配分順序の優先制御を行うことによって複数の実ＣＰＵを割り付けた複数のゲストＶMの配分処理において指定された配分

14

量の通りに正確に配分処理を繰り返すことができ、実ＣＰＵ資源を有効利用した配分処理による仮想計算機システムの動作ができる。

【００６５】また、ＣＰＵ配分を受けたゲストＶMが待ち状態となっても次のＣＰＵ配分では元の配分順序に戻ることでゲストＶMの待ち状態の発生により配分を乱されることなく実ＣＰＵ資源を有効に利用することができる。

【図面の簡単な説明】

【図１】本発明の原理説明図

【図２】本発明のハードウェア構成を示した実施例構成図

【図３】本発明によるＣＰＵ資源の配分処理の一実施例を示した説明図

【図4】本発明によるＣＰＵ資源の配分処理を示したフローチャート

【図5】本発明による配分比が不均等な場合の配分処理の順番を示した説明図

【図6】図5 の配分処理をＣＰＵの割当時間で示した説明図

【図7】本発明で全てのゲストＶMが動作可能な時のＣＰＵ配分の順番を示した説明図

【図8】配分中に待ち状態が起きた場合の配分処理を示した説明図

【図9】従来の単一の実ＣＰＵを用いた仮想計算機システムの説明図

【図１０】図9 の単一実ＣＰＵにおけるＣＰＵ配分処理を示したフローチャート

【図１１】従来の配分比に従った配分順を示した説明図

【図１２】従来の複数の実ＣＰＵを用いた仮想計算機システムの説明図

【図１３】複数の実ＣＰＵを用いた従来の均等配分の処理説明図

【図１４】複数の実ＣＰＵを用いた従来の不均等配分の処理説明図

【図１５】図１４の不均等配分をＣＰＵ割当時間で示した説明図

【図１６】図１４の不均等配分の繰り返して飽和状態に至った時の配分比の指定量と実際の割当量を示した説明図

【符号の説明】

１０－1 ～１０－n：仮想計算機（ゲストＶM）

１２－1，１２－2：ＣＰＵ（実ＣＰＵ）

１４：制御プログラム（ＣＰ）

１６：ＣＰＵ割当情報域（処理制御ブロック；ＰＣＢ）

１８：レディキュー

２０：動作カウンタ

２２：配分順序制御手段

２４：ウェイトキュー

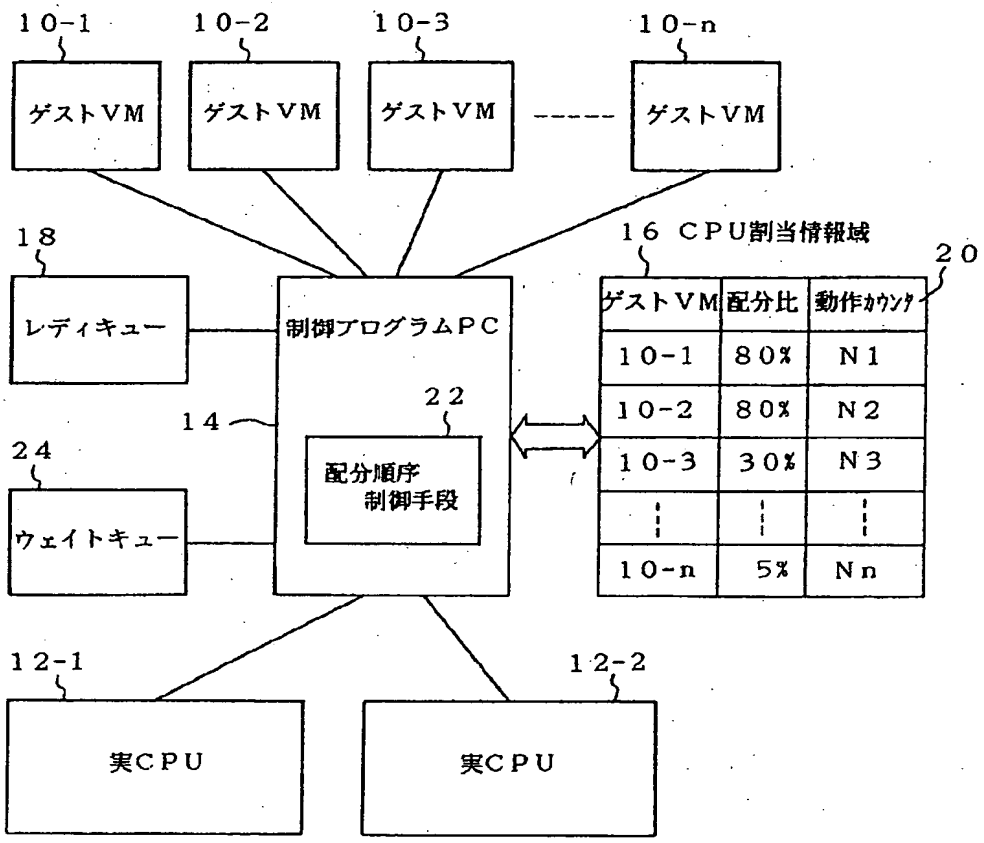
２６：主記憶制御装置（ＭＣＵ）

15　　16

２８：主記憶装置(MS U)
３０：オペレーティングシステム(OS)
３２：チャネルプロセッサ(CHP)
３４：デバイスバス
３６－１，３６－２：デバイスインタフェース
３８：磁気ディスク装置

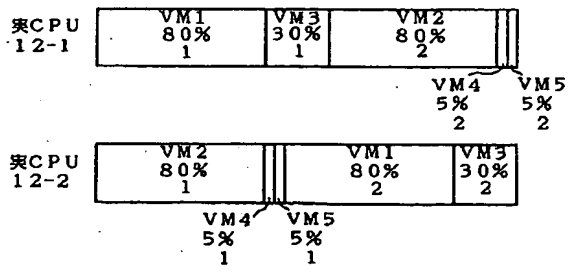
【 図1 】

本発明の原理説明図



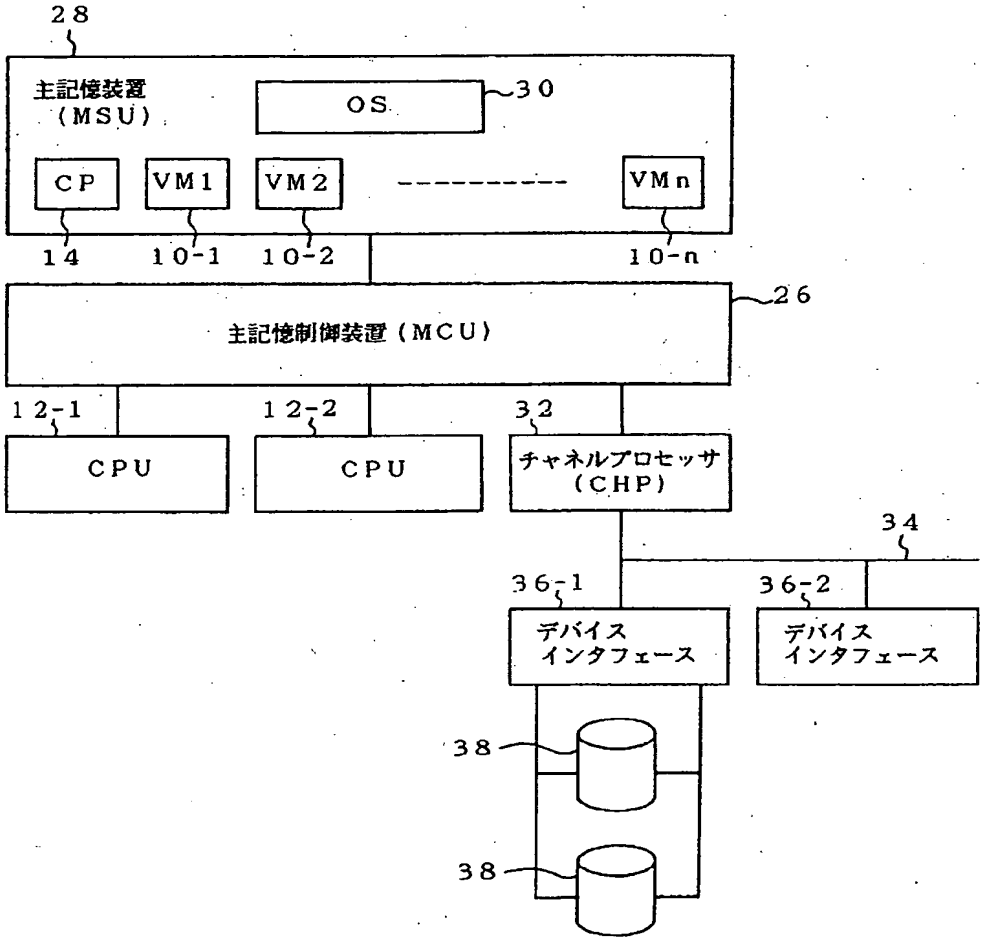
【 図6 】

図5の配分処理をＣＰＵの割当時間で示した説明図



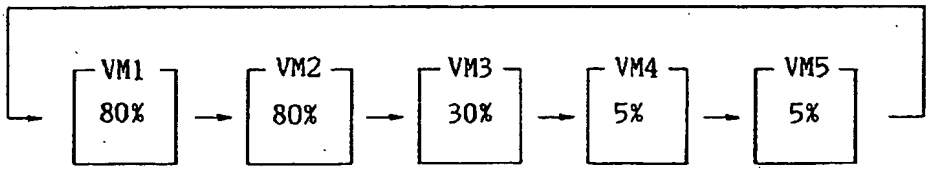
【図2】

本発明のハードウェア構成を示した実施例構成図



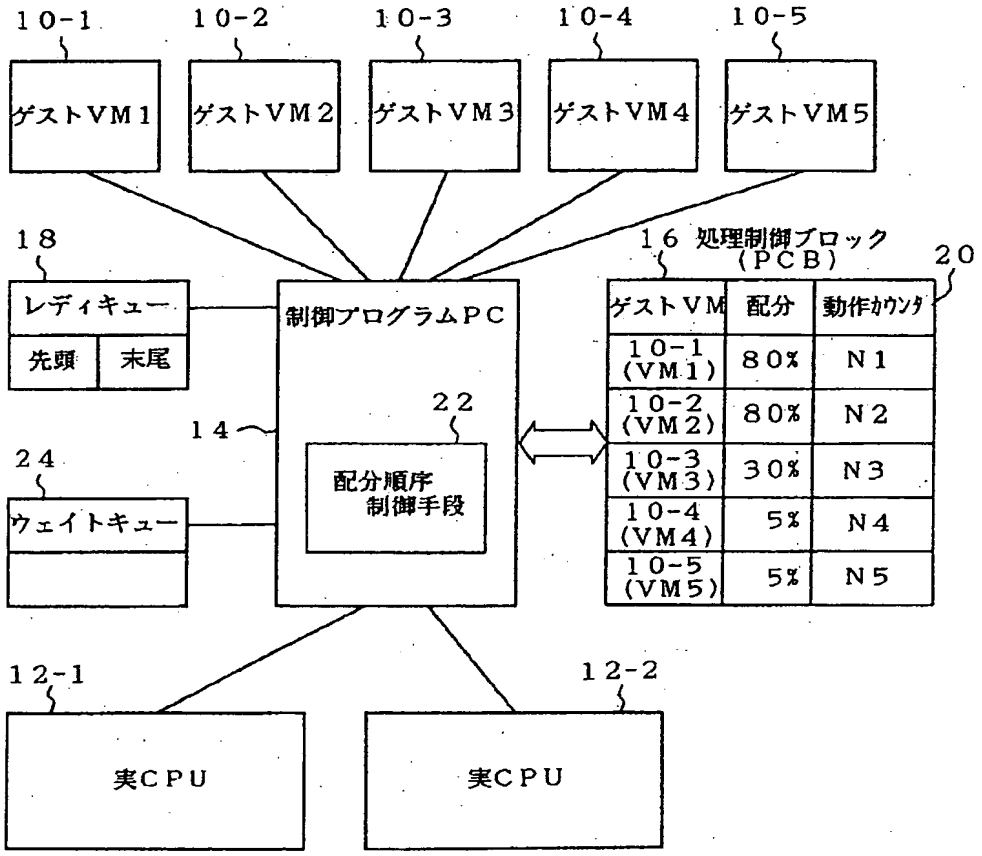
【図7】

本発明で全てのゲストＶＭが動作可能な時のＣＰＵ配分の順番を示した説明図



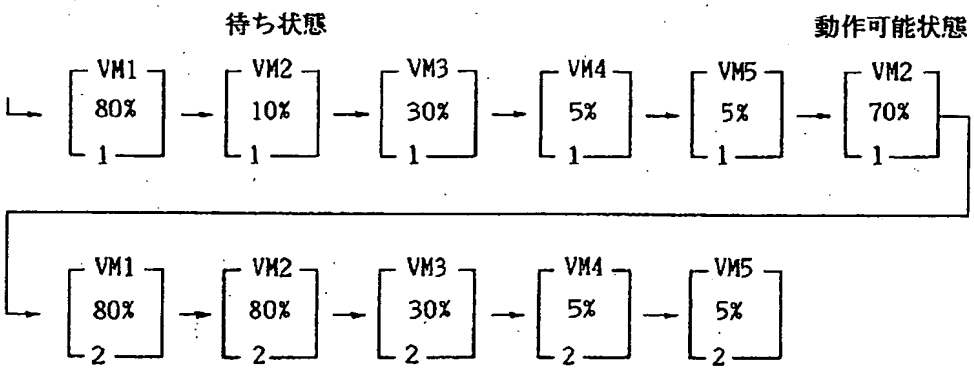
【図3】

本発明によるＣＰＵ資源の配分処理の一実施例を示した説明図



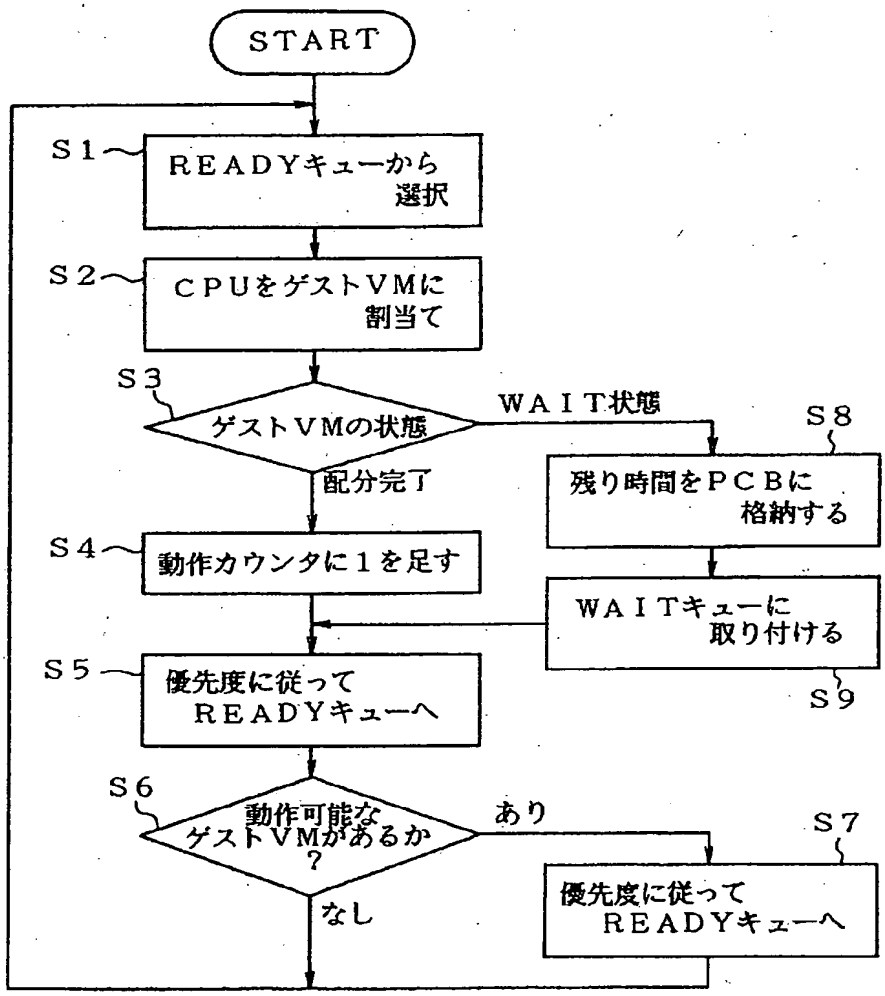
【図8】

配分中に待ち状態が起きた場合の配分処理を示した説明図



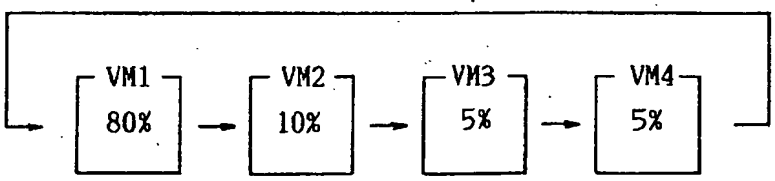
【図4】

本発明によるＣＰＵ資源の配分処理を示したフローチャート

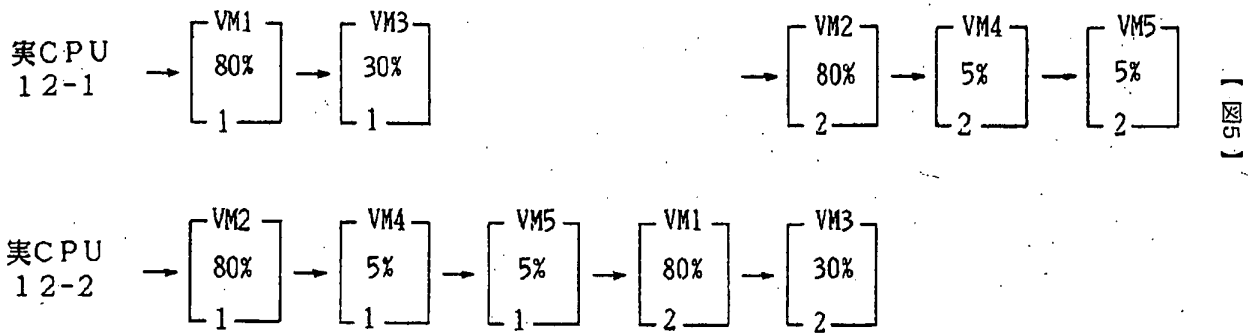


【図11】

従来の配分比に従った配分順を示した説明図

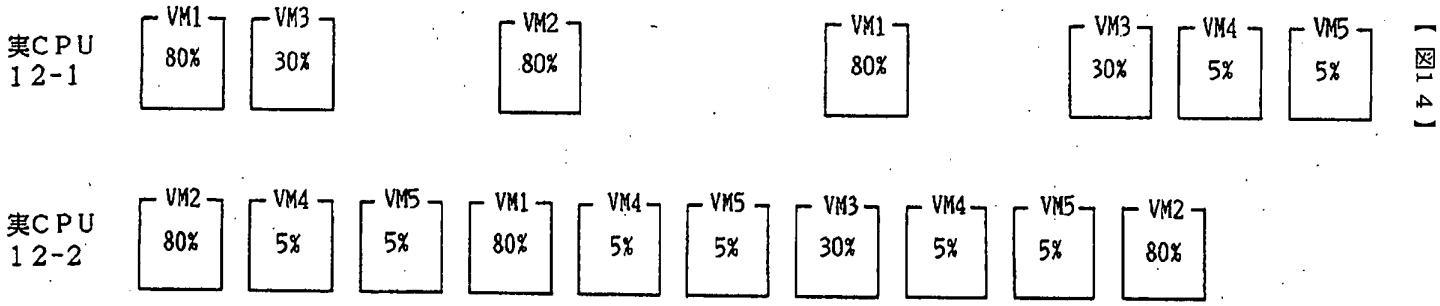


本発明による配分比が不均等な場合の配分処理の順番を示した説明図



【図5】

複数の実ＣＰＵを用いた従来の不均等配分の処理説明図

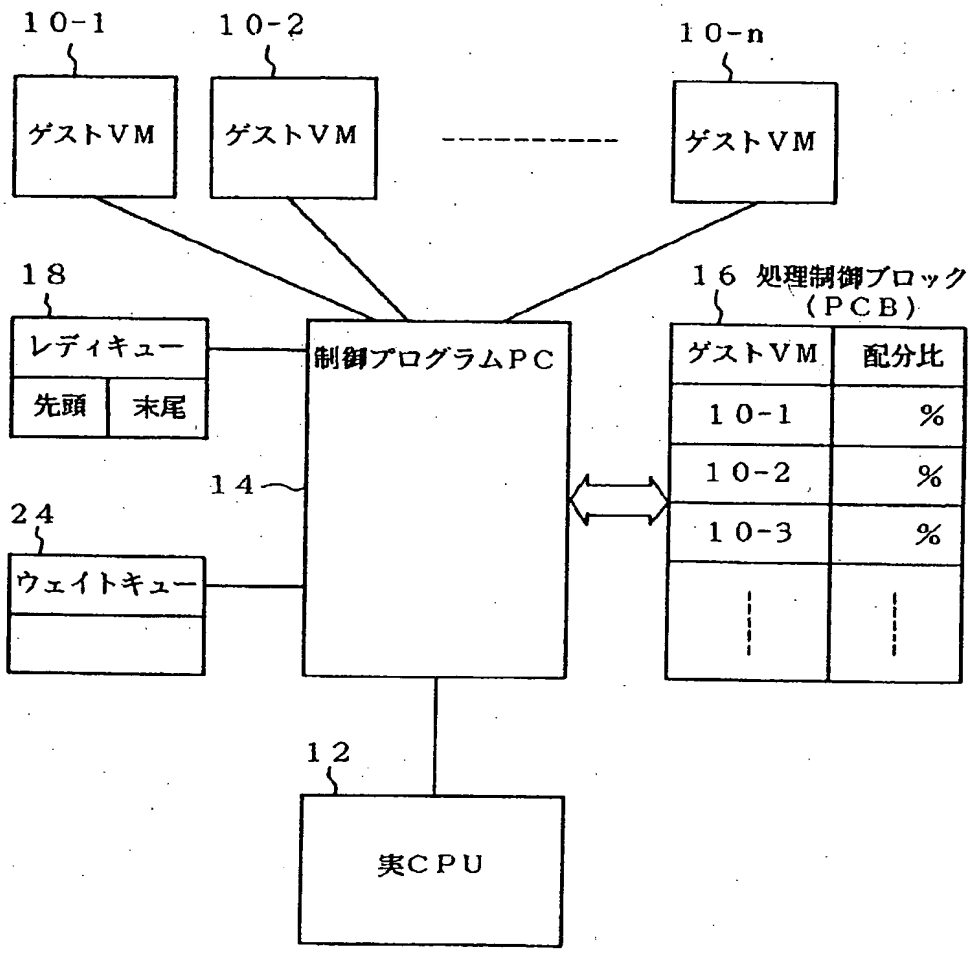


【図14】

特開平5 −324361

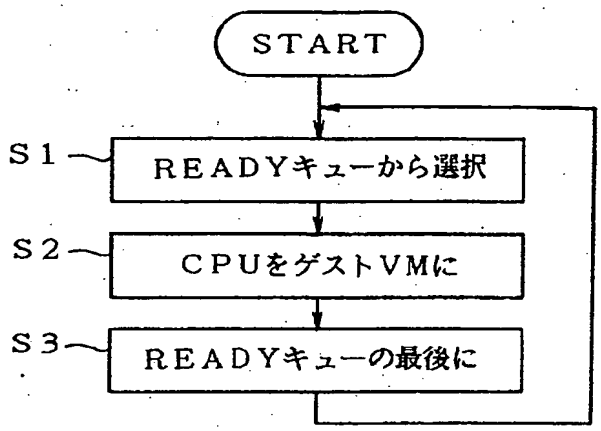
【 図9 】

従来の単一の実ＣＰＵを用いた仮想計算機システムの説明図



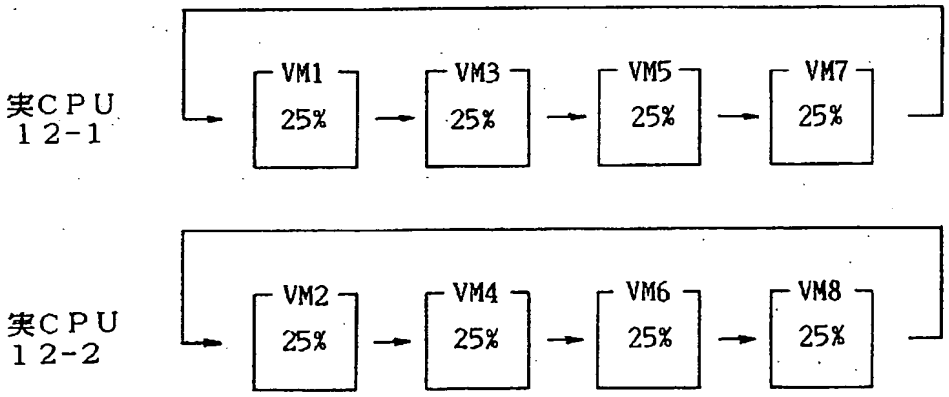
【図１０】

図９の単一実ＣＰＵにおけるＣＰＵ配分処理を示したフローチャート



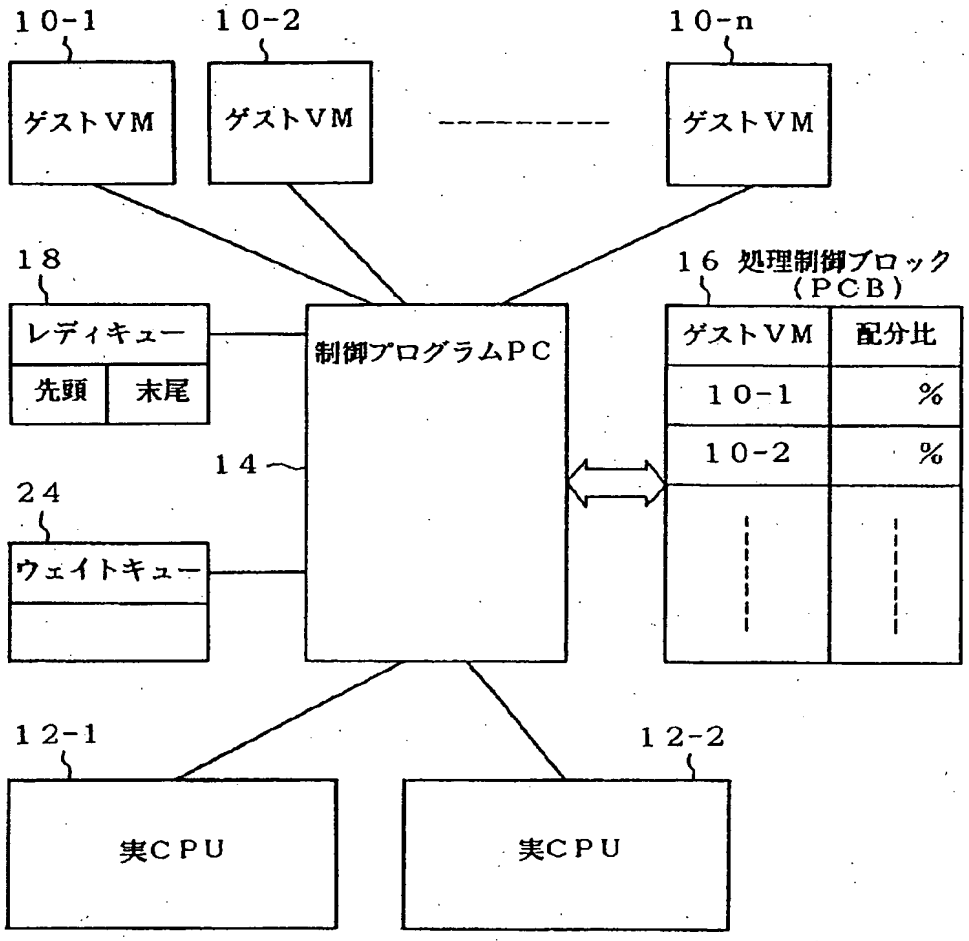
【図１３】

複数の実ＣＰＵを用いた従来の均等配分の処理説明図



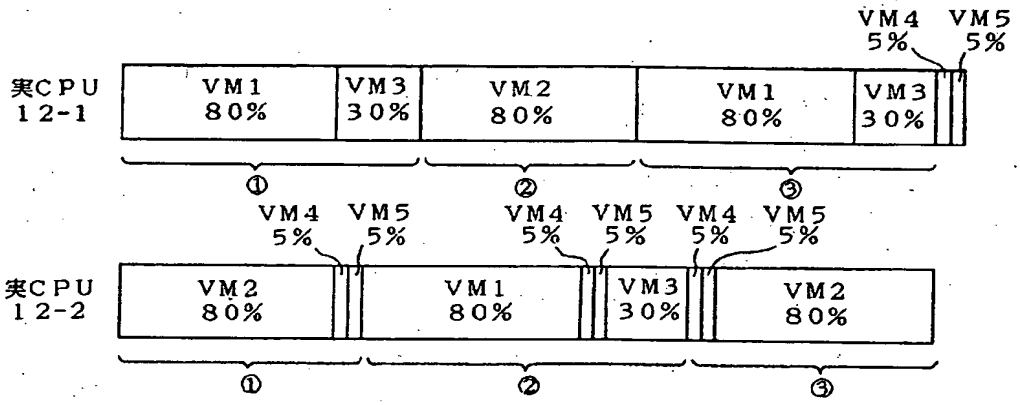
【図12】

従来の複数の実ＣＰＵを用いた仮想計算機システムの説明図



【 図15 】

図14の不均等配分をCPU割当時間で示した説明図



【 図16 】

図14の不均等配分の繰り返して飽和状態に至った時の配分比の指定量と実際の割当量を示した説明図

VM名	割当量	指定量
VM1	78	80
VM2	78	80
VM3	28	30
VM4	7	5
VM5	7	5